Explicit Integration of Extremely-Stiff Reaction Networks: Asymptotic Methods

M. W. Guidry[1,2,3], R. Budiardja[1,2,3], E. Feger[1], J. J. Billings[3], W. R. Hix[1,2,3], O. E. B. Messer[4,1], K. J. Roche[5,6], E. McMahon[1], and M. He[7,1]

[1]Department of Physics and Astronomy, University of Tennessee, Knoxville, TN 37996-1200, USA
[2]Physics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA
[3]Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA
[4]National Center for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
[5]Pacific Northwest National Laboratory, Richland, WA 99352, USA.
[6]Department of Physics, University of Washington, Seattle, WA 98195, USA.
[7]Department of Mathematics, Shanghai Jiao-Tong University, Shanghai, PRC

E-mail: guidry@utk.edu

Abstract. We show that, even for extremely stiff systems, explicit integration may compete in both accuracy and speed with implicit methods if algebraic methods are used to stabilize the numerical integration. The required stabilizing algebra depends on whether the system is well-removed from equilibrium or near equilibrium. This paper introduces a quantitative distinction between these two regimes and addresses the former case in depth, presenting explicit asymptotic methods appropriate when the system is extremely stiff but only weakly equilibrated. A second paper [1] examines quasi-steady-state methods as an alternative to asymptotic methods in systems well away from equilibrium and a third paper [2] extends these methods to equilibrium conditions in extremely stiff systems using partial equilibrium methods. All three papers present systematic evidence for timesteps competitive with implicit methods. Because an explicit method can execute a timestep faster than an implicit method, algebraically-stabilized explicit algorithms might permit integration of larger networks than have been feasible before in various disciplines.

PACS numbers: 02.60.Lj, 02.30.Jr, 82.33.Vx, 47.40, 26.30.-k, 95.30.Lz, 47.70.-n, 82.20.-w, 47.70.Pq

Keywords: ordinary differential equations, reaction networks, stiffness, reactive flows, nucleosynthesis, combustion

1. Introduction

In many scientific and technical contexts one encounters phenomena that may be modeled by fluxes transferring population between sources and sinks for various species. Examples

include kinetic processes that modify abundances and transfer energy in atomic, molecular, and nuclear systems; geochemical, climate, and other environmental systems; electrical circuits; economic models; and population dynamics. Terminology varies but let us refer generically to these sources and sinks as *boxes,* and term the resulting systems of boxes connected by fluxes *reaction networks.* Such systems are commonly modeled by a coupled set of differential equations that describe a continuous flow of population through the boxes.

The reaction network is often classified as a *stiff system*, which we shall define to be a system of equations containing multiple timescales ranging over many orders of magnitude [3, 4, 5, 6]. Most physical systems involve important processes operating on very different timescales, so realistic problems tend to be at least moderately stiff. Some, such as those encountered in many astrophysics applications, are extremely stiff, with fastest and slowest timescales in the problem differing by as much as 10–20 orders of magnitude. In stiff systems the timestep constraints are set by numerical stability requirements rather than accuracy considerations. Hence, explicit numerical integration of stiff systems is usually impractical because the maximum stable timestep is far too small for efficient solutions (see, for example, Refs. [5, 6]). This is commonly addressed by employing implicit or semi-implicit stiff solvers that are stable, but that require time-consuming iterative matrix solutions.

A given box in a reaction network often is connected strongly only to a few other boxes. For example, the explosive burning conditions encountered in astrophysical novae, X-ray bursts, or supernovae may require reaction networks with hundreds to thousands of nuclear isotopes. Yet individual isotopes are typically connected directly to other isotopes through (at most) $\sim$10 reactions of consequence, and under many conditions no more than 2–3 reactions are important for a given isotope. Such restrictions on the direct box reaction coupling imply that the matrices appearing in the iterative implicit solution are *sparse.* Although various methods are available to deal with sparse matrices, in practice many codes for solving large reaction networks have not exploited sparseness in particularly effective ways.

For example, in astrophysical calculations with implicit solvers in large networks (say $\sim$ 150 species or more), one finds often that greater than 90% of the processor time is consumed in matrix operations [7, 8]. Efficient algorithms exist for the required matrix algebra (with incremental improvements in them over time), but the matrix nature of the core problem implies that the time required for implicit solution grows non-linearly with the size of the network. In typical working codes for large-scale applications, increasing the size of the network increases the time for solution, often quadratically, sometimes as much as cubically, until there are enough boxes in the network to justify the overhead of sparse-matrix methods with more favorable scaling. In applications in thermonuclear networks, for example, it is often found that the overhead required to implement sparse-matrix iterative solutions is not justified until there are several hundred boxes in the network. Thus, many present implicit stiff-network algorithms do not scale very gracefully to larger networks.

We are primarily interested in the most ambitious applications of large networks, where the reaction network is only a portion of a larger problem. Let us take as representative astrophysical thermonuclear reaction networks, where a proper description of the overall problem typically requires multi-dimensional hydrodynamics or radiation hydrodynamics

coupled tightly to a large thermonuclear reaction network. The hydrodynamical evolution controls the conditions in the network such as temperature and density, and the network influences the hydrodynamic evolution strongly through energy production and modification of composition variables. As a consequence of the limitations discussed in the preceding paragraphs, the solution of large networks by the usual approaches is time-consuming and few calculations have attempted to couple the element and energy production strongly to the hydrodynamics with a network of realistic complexity. The most ambitious approaches use very small networks, perhaps tuned empirically to get critical quantities like energy production correct on average, coupled to the hydrodynamical simulation. In many calculations even this is not done and the network is replaced entirely by parameterization. Then a more complete network is run in a separate "post-processing" step, where fixed hydrodynamical profiles computed in the hydrodynamical simulation with the small network are used to specify the variation of thermodynamic variables such as temperature and density with time.

Astrophysical thermonuclear networks have been used for illustration, but many problems of scientific and technical interest exhibit similar complexity. Examples include astrochemical kinetics, where one must model large chemical evolution networks in contracting molecular clouds, or combustion chemistry, where chemical burning networks are strongly coupled to simulations of the dynamics of the air and fuel mixture. Physically-realistic networks in such contexts would often be quite large. In combustion of larger hydrocarbon molecules or studies of soot formation, hundreds to thousands of reacting species undergoing as many as 10,000 reactions may be encountered [6], and in supernova explosions hundreds to thousands of nuclear isotopes with tens of thousands of reaction couplings make non-zero contributions [8]. For such cases one finds that current techniques do not allow for a coupling of realistic reaction networks to the full dynamics of the problem and often severely truncated or highly schematic reaction networks have been used in even the most realistic simulations.

2. Reaction Networks in the Context of Larger Problems

To be definite, we shall assume that the coupling of reaction networks is done using operator splitting, where the hydrodynamical solver is evolved for a numerical timestep holding network parameters constant, and then the network is evolved over the time corresponding to the hydrodynamical timestep holding the new hydrodynamical variables constant. This places two basic constraints on methods:

(i) At the end of each hydrodynamical timestep the network must be advanced with new initial conditions. Thus, algorithms must be capable of rapid initialization and must not depend in a complex way on conditions from previous time intervals.

(ii) With modern processors, existing algorithms are reasonably adequate for many post-processing calculations. In contrast, for the operator-split, parallel processing environment that is our interest here, solution of the network over a hydrodynamic

timestep must be fast enough that it does not require time substantially larger than that for the hydrodynamical solution.

Let us elaborate further on this second point. If a single processor were used to calculate both the hydrodynamical evolution and the network evolution in one hydrodynamical zone, the network evolution over a hydrodynamical timestep interval must be fast enough to not slow the calculation by too much relative to the hydrodynamical evolution alone. If we take the point of view that we are willing to tolerate longer compute times in the interest of a much more realistic calculation, but not longer by orders of magnitude, we estimate that the network must be capable of evolving over the time interval corresponding to the hydrodynamical timestep in roughly a second or less wall clock time.

We take the multidimensional, adaptive-mesh, explicit hydrodynamical FLASH code [9] applied to Type Ia supernova simulations on large parallel systems as representative. The explicit hydrodynamical timestep will be limited overall by the Courant time (roughly, because stability requires a hydrodynamical timestep not larger than the sound-crossing time for the zone), and more stringently in zones of rapid burning where temperature and density may be changing rapidly. In current Type Ia supernova simulations the Courant time would typically be 10^{-4} s or smaller over most of the grid for the timescale relevant for the main part of the explosion, with rapid nuclear burning and associated temperature changes limiting the hydrodynamical timestep to 10^{-8}–10^{-10} s for some ranges of times. For qualitative estimates, let us take as representative that a typical network integration for a single hydrodynamical timestep will be over an interval $\sim 10^{-8}$ s during the time of strong burning and $\sim 10^{-6}$ s over much of the approach to equilibration after strong burning.

In FLASH, many spatial zones will be assigned to a single MPI rank on a parallel system. Therefore, in the absence of node-level parallelism (for example, with OpenMP), the network must be capable of calculating a number of hydrodynamic time intervals in a second or less if we wish to calculate an independent network for each zone. Let us take for estimation purposed that we wish to be able to reliably integrate 1000 independent networks over a time interval of say 10^{-6} seconds on a single processor in one second wall clock time, with each network containing several hundred isotopes. This places extremely strong startup and speed constraints on the required network. The explicit algorithms discussed here are capable of perhaps 10^4 network timesteps per second on a single processor with present technology for a network with ~ 150 isotopes, so our goals require a network algorithm that can integrate a time interval of order 10^{-6} seconds in no more than $\sim 10-100$ timesteps, implying average stable and accurate timesteps at least as large as $10^{-2}-10^{-3}$ times the elapsed integration time for the corresponding hydrodynamical evolution. Figure 1 illustrates.

Such large timesteps are often possible with implicit and semi-implicit algorithms, but those methods are inefficient at computing each timestep; explicit methods can compute a timestep efficiently, but timesteps this large are unthinkable with a normal explicit algorithm because they would be unstable in most realistic situations. In this and the other two papers [1, 2] of this series we shall demonstrate stabilization methods for explicit integration that realize such competitive integration timesteps in a variety of examples. Thus we shall reopen

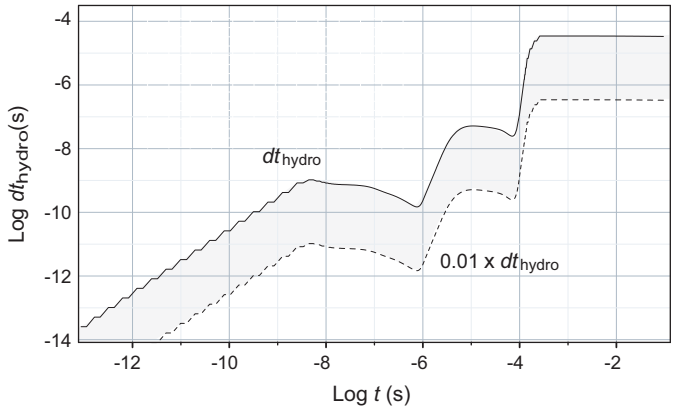

Figure 1. Hydrodynamical timesteps in a typical Type Ia supernova simulation (solid upper curve). The calculation corresponds to a single zone integrated with the Flash hydrodynamical code [9], operator-split coupled to a 150-isotope network using the explicit asymptotic method described below. Initial conditions were equal mass fractions of ^{12}C and ^{16}O, with an initial temperature of 3×10^9 K and initial mass density of 10^7 g cm^{-3}. The shaded region between the two curves represents the range of timesteps an explicit network calculation must take to be able to integrate 100-1000 zones on a single processor over one operator-split hydrodynamical timestep in less than about one second elapsed time on modern processors. The hydrodynamical timestep lies roughly in the range $0.1t$–$0.001t$, where t is the elapsed time, over most of the range of integration. To maintain network timesteps within the band for each operator-split hydrodynamical timestep, we see that the algorithm must be capable of taking stable and accurate network timesteps approximately in the range $0.01t$–$0.0001t$ over the entire range of hydrodynamical integration.

the discussion of whether explicit methods, with their faster computation of timesteps and more favorable scaling with network size, are practical for large, stiff networks.

3. Stiffness in Reaction Networks

The general task is to solve efficiently N coupled ordinary differential equations

$$\begin{aligned}\frac{dy_i}{dt} &= F_i(y,t) = \sum_j F_{ij}(t) \\ &\equiv F_i^+(t) - F_i^-(t) = F_i^+(t) - k_i(t)y_i(t) \end{aligned} \tag{1}$$

subject to appropriate boundary conditions. In this expression, the $y_i (i = 1 \ldots N)$ describe the dependent variables (typically measures of abundance), t is the independent variable (the time in our examples), the fluxes between species i and j are denoted by F_{ij}, and $k_i(t)$ is the effective rate for all processes depleting the species i. The sum for each variable i is over all species j coupled to i by a non-zero flux F_{ij}, and for later convenience we have decomposed the flux into a component F_i^+ that increases the abundance of y_i and a component $F_i^- = k_i y_i$ that depletes it. For an N-species network there will be N such equations in the population variables y_i, generally coupled to each other because of the dependence of the fluxes on the different y_j. (For notational simplicity we will not always display the i index explicitly on the

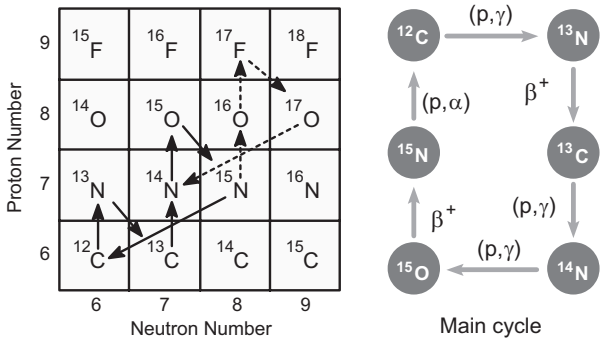

Figure 2. The CNO cycle. On the left side the main branch of the cycle is illustrated with solid arrows and a side branch is illustrated with dashed arrows. On the right side, the main branch of the CNO cycle is illustrated in more detail.

right side in our equations). The variables y_i are typically proportional to a number density for the species i. To keep the discussion general the variable $y_i(t)$ will be used in most of our equations, but for the specific astrophysical examples that follow we shall replace the generic population variables y_i with the mass fraction X_i, which satisfies

$$X_i = \frac{n_i A_i}{\rho N_{\rm A}} \qquad \sum_i X_i = 1, \tag{2}$$

where $N_{\rm A}$ is Avogadro's number, ρ is the total mass density, and A_i is the atomic mass number and n_i the number density for the species i.

3.1. Example: Stiffness and Stability in the CNO Cycle

The carbon–nitrogen–oxygen (CNO) cycle that powers main-sequence stars more massive than the Sun provides a graphic illustration of stiffness in a relatively simple system of large physical significance. The CNO cycle is displayed in Fig. 2. We shall illustrate by considering the primary part of the CNO cycle illustrated on the right side of the figure. If the thermonuclear network corresponding to the main part of the CNO cycle is integrated under typical CNO cycle temperature and density conditions by explicit forward Euler methods using standard rates [10] for the reactions and constant timesteps, the integration is stable for timesteps less than or equal to 285.7 seconds, but becomes catastrophically unstable for a timestep of 285.8 seconds or more. This instability threshold is *precisely two over the fastest rate* for the transitions in the network, which corresponds to the β-decay of ^{15}O to produce ^{15}N.

We now show that this instability for forward differencing of the CNO cycles arises because rapidly-decreasing small populations can become negative in an explicit integration if the timestep is too large. These negative populations export unphysical negative population that can destabilize the system because they can lead to exponentially growing solutions in small components that ultimately couple to the larger components. Let us elaborate through the use of a simple model illustrated in Figs. 3–5, which will generalize a discussion that may be found in Ref. [6].

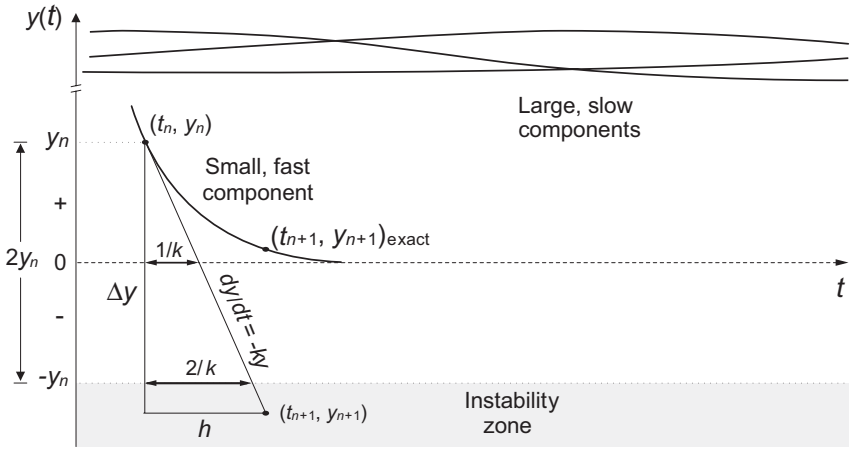

Figure 3. One common origin of stiffness instability. For simplicity in discussion the timescale h is presumed to be sufficiently short that the large, slow components remain essentially unchanged over a time equal to many times h. The explicit integration for the small, fast component diverges if $y_{n+1} \leq -y_n$, such that y_{n+1} lies in the shaded region. This divergence is exponential for longer times and quickly propagates to the larger slow components, destabilizing the entire network. The illustration of the slow components is schematic; in typical stiff systems their timescales may be many orders of magnitude longer than those for fast components and often they are larger in value by many orders of magnitude.

3.2. A Simple Model for One Form of Stiffness

In Fig. 3 we assume a coupled system in which there are many components varying on a relatively long timescale (illustrated schematically by the curves at the top) and a single component that is small and exponentially decreasing on a much faster timescale, with a typical behavior $y(t) = \exp(-kt)$ so that $dy(t)/dt = -ky(t)$. Let us consider timesteps that remain small compared with the larger timescales in the system (so that for a single step we are in an approximately adiabatic situation and the populations varying on long timescales may be considered frozen), but comparable to or larger than the timescale set by $1/k$.

To advance the solution for the fast component from t_n to t_{n+1} by the explicit Euler method, we take a timestep h and extrapolate the solution using the derivative $dy/dt = -ky$ evaluated at (t_n, y_n). The triangle of height Δy in Fig. 3 summarizes, where (t_{n+1}, y_{n+1}) represents the numerical solution and $(t_{n+1}, y_{n+1})_{\text{exact}}$ is the exact solution. From this construction we see that [6]

(i) For $h < 1/k$, the forward Euler approximant for step $n+1$ will yield a value of y_{n+1} that lies between 0 and y_n, so that $|y_{n+1}| < |y_n|$.

(ii) For $1/k < h < 2/k$, the sign of y_{n+1} will be negative but again $|y_{n+1}| < |y_n|$.

(iii) For $h \geq 2/k$ the sign of y_{n+1} will be negative and $|y_{n+1}| \geq |y_n|$.

But for the forward Euler approximation to $y(t) = \exp(-kt)$ we have $y_{n+1} = (1 - hk)y_n$, so that by iterating for m successive steps of fixed size h (with mh still considerably less than the

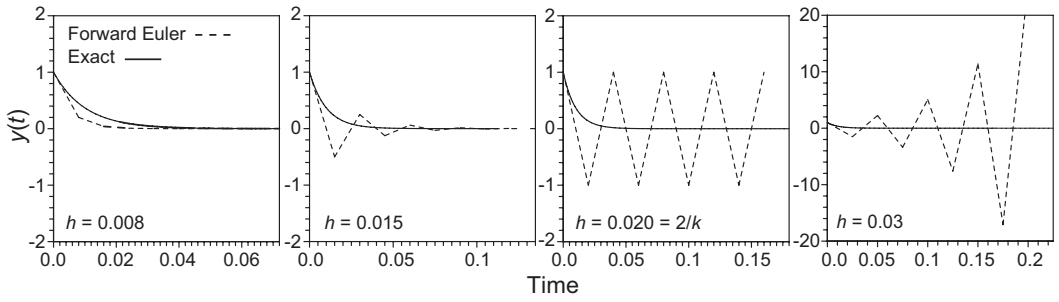

Figure 4. Behavior of $\exp(-kt)$ with $k = 100$ for increasing explicit timesteps h. Generally for $h < 2/k$ the solution converges to the correct value of zero but for $h > 2/k$ the solution diverges to $\pm\infty$ under successive iterations. Note that the vertical scale has been increased by a factor of 10 in the rightmost figure.

longer timescales in the system, so that the adiabatic approximation remains valid), we obtain

$$y_{n+m} = (1 - hk)^m y_n. \tag{3}$$

This converges toward the correct value of zero at larger times only if the product hk lies between zero and two, implying that the maximum value of h that yields a convergent solution is bounded by $h < 2/k$. Thus, in Fig. 3 the maximum stable value of h is less than $2/k$ and any point (t_{n+1}, y_{n+1}) extrapolated by forward Euler integration that lies in the shaded zone (that is, $y_{n+1} \leq -y_n$) will be unstable under the iteration (3), diverging to infinity rather than converging to the correct value of zero. Because the small, fast components are coupled to the other components of the network, this divergence will quickly destabilize the entire network. Let us illustrate with a concrete example. Figure 4 corresponds to forward Euler solution of $y_{n+1} = (1 - hk)y_n$ with $k = 100$ and timesteps ranging between $h = 0.008$ and 0.03. In this example, we see that

(i) For $h < 2/k$, the solution converges to zero.

(ii) For $h = 2/k$, the solution oscillates between positive and negative values of the same absolute value and neither converges nor diverges.

(iii) For $h > 2/k$, the solution diverges to $\pm\infty$ under successive iterations, with the divergence exhibiting exponential behavior for larger times.

Figure 5 applies the preceding model of the origin of the stiffness instability to the CNO cycle. As noted earlier, (for the parameters assumed) ordinary explicit Euler integration is highly unstable for a timestep larger than 285.7 seconds. This critical timestep is exactly two divided by the fastest rate parameter in the system, which is that for the β-decay of ^{15}O to ^{13}N. Expressing the network in matrix form and diagonalizing indicates that the origin of this instability lies in eigenvalues that exceed unity in absolute value if $h > 2/k_{\max}$. Generalizing the discussion associated with Eq. (3), for a coupled set of equations a finite-difference iteration for m steps entails a matrix raised to the power m applied to the original y vector. This is guaranteed to converge only if no eigenvalue of the matrix exceeds unity in magnitude.

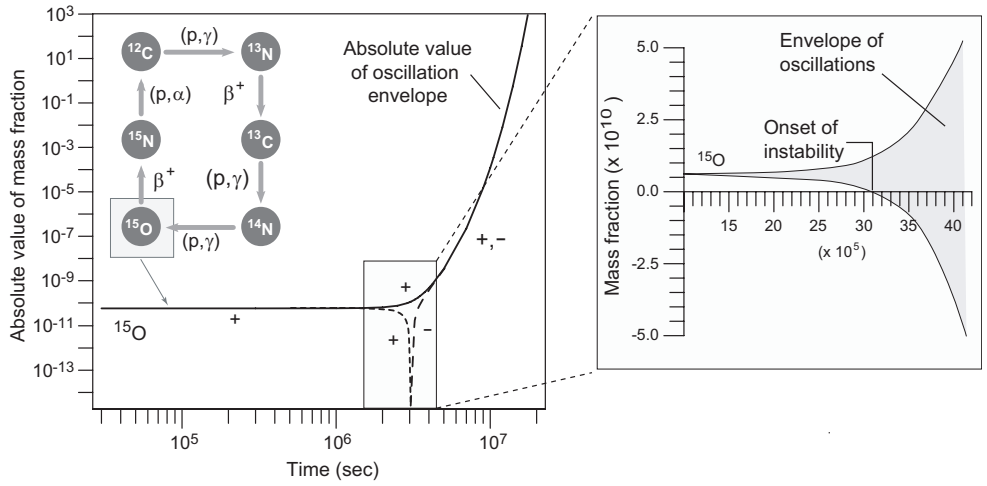

Figure 5. Origin of the stiffness instability for explicit Euler-method integration of the CNO cycle main branch (shown as an inset). If the timestep is too large the mass fraction of ^{15}O, which normally is of order 10^{-10} or smaller under the conditions assumed here, can become negative. This unphysical condition is unstable and triggers an exponential runaway that quickly crashes the entire network. On the left side, the *absolute value* of the envelope of the oscillating solution is plotted on a log scale, with the sign indicated adjacent to different parts of the curves. The right side shows a blowup of the region where the mass fraction begins to oscillate into negative values, now plotted on a linear scale to show clearly the envelope of the diverging oscillating solution. This figure is a generalization of Fig. 3 to a more complex network.

Less abstractly, Fig. 5 indicates that the origin of this instability for standard explicit integration is the tendency of the ^{15}O population to become negative for large explicit timesteps. This is a more complex system than the previous simple decaying-exponential example because the ^{15}O population is depleted by the β-decay but also replenished by the (p,γ) proton capture reaction on ^{14}N (see Fig. 2 and the inset diagram on the left side of Fig. 5). Nevertheless, we see that the origin of the stiffness instability is very similar to that illustrated in the preceding simple example: a small box population becomes unphysically negative because of taking too large a timestep, so in the next timestep the offending box exports an unphysical flux of negative population, triggering a divergence that rapidly compromises the entire network.

Motivated by the properties of stiffness illustrated in the previous examples, we first introduce a rather crude approximation for stabilizing explicit integration in stiff networks. Although we shall discuss much better algorithms after that, it is instructive that even this simple approximation, by removing a primary source of stiffness, leads to an explicit algorithm that is usable in large, realistic, extremely-stiff networks.

4. Flux-Limited Forward Difference Algorithm

The results of the previous section indicate that one form of stiffness instability is generated by box populations that become very slightly negative and that these destabilize the network

when they are exported as fluxes to other boxes (effectively because this anomalous sign for the box population turns what should be exponential decay into exponential growth). Thus, we invoke a simple flux-limiting prescription that if the population of a box becomes negative we do not change the population itself (which would quickly violate conservation of probability) but we suppress all export of that negative population to other boxes. Thus, in the course of a calculation no flux is permitted out of boxes that have negative populations until their populations again become positive because of flux into them. Formally, we require for all computed outgoing fluxes that $F_{ij} \rightarrow \max(F_{ij}, 0) \equiv \tilde{F}_{ij}$. In shorthand, we refer to this as *suppression of negative flux,* and refer to the resulting algorithm as the *flux-limited forward difference (FLFD) algorithm.* In its simplest implementation, which we shall illustrate here, we use the (explicit) forward Euler method supplemented by the flux-limiting prescription, but the same approach can be applied to higher-order forward differencing.

Because we do not alter populations but only restrict their flow by this algorithm, it conserves probability. We may expect that some populations in the network will now be in error because of the flux-suppression criterion. However, such effects tend to involve the smallest populations (because they are the ones most easily made negative by the numerical error), so we may expect that this approximation could be a good one for the larger populations, with the error concentrated in the smallest populations. For large networks coupled to hydrodynamical evolution it is the changes in the larger populations that dominate energy production and concentration changes, so errors in the smaller populations are largely irrelevant. Therefore, an algorithm that removes stiffness by preferentially concentrating errors in the smallest populations may be a very usable one. In the next section we test this idea on a realistic thermonuclear network.

5. Flux-Limited Network Solutions under Nova Conditions

We illustrate the flux-limited forward difference algorithm by application to astrophysical thermonuclear networks under nova conditions.

5.1. Hot CNO Burning

Some representative isotopic abundances under nova (hot CNO cycle) conditions calculated using the FLFD algorithm are shown in Fig. 6. An initial isotopic abundance distribution enriched in heavy elements has been assumed [12], and reaction rates from the REACLIB library [10] have been used. Figure 6(a) displays some representative populations, with the results of a standard implicit calculation [11] shown as dashed lines and FLFD calculations shown as symbols. An adaptive timestep was used in the FLFD integration, with the timestep adjusted to keep the populations transferred between boxes in a timestep for some key populations within a prescribed range. This timestep is *much* larger than would be stable in a standard explicit integration, as we shall discuss further in §5.2. Note the very good agreement between implicit and explicit flux-limited methods over six orders of magnitude in the mass fractions in Fig. 6(a). In a realistic coupling of such a network to hydrodynamics in

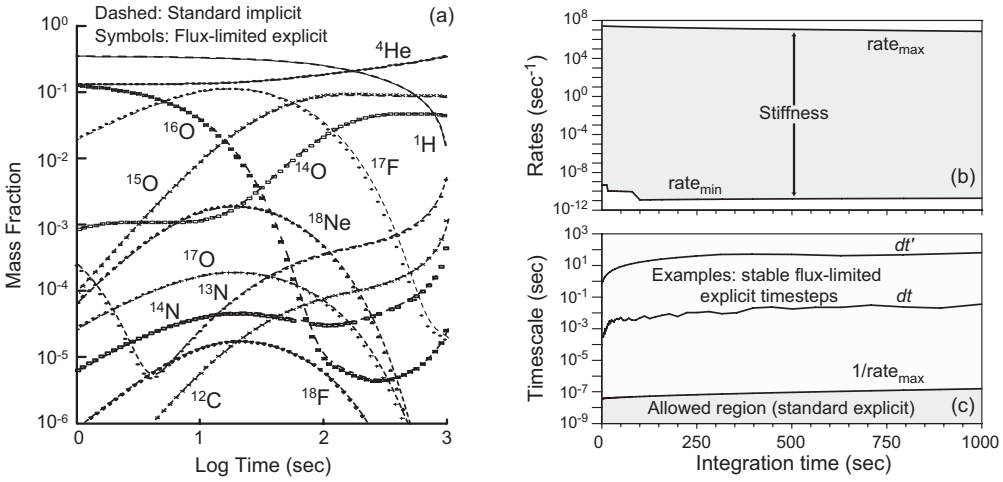

Figure 6. (a) Some representative isotopic abundances under nova (hot CNO) conditions calculated using the explicit FLFD algorithm and compared with a calculation using the standard implicit solver Xnet [11]. A constant temperature of $T = 0.25 \times 10^9$ K and constant density $\rho = 500\,\mathrm{g\,cm^{-3}}$ were assumed. The explicit network contained 145 isotopes, with 924 non-zero couplings. (b) Rates and timescales characteristic of a FLFD nova simulation. Conditions as for part (a) but with a larger reaction library: 896 isotopes with 8260 couplings were included (though only several hundred isotopes were populated significantly). Extremal rates plotted are restricted to those involving non-zero fluxes. (c) Comparison of maximum stable timestep [$\simeq 1/\mathrm{rate_{max}}$ from part (b)] possible for a standard explicit integration with much larger stable timesteps dt and dt' for some representative explicit FLFD integrations.

a nova simulation, isotopes with mass fractions smaller than 10^{-2}–10^{-3} would likely be not very important to the hydrodynamical evolution.

5.2. Stiffness and Stability

Figure 6(b) displays the fastest and slowest rates entering a representative FLFD nova simulation as a function of time. The difference of some 18 orders of magnitude between the fastest and slowest rates at any timestep is an indication that this is an extremely stiff system. For standard explicit algorithms, the largest timestep permitted by stiffness stability criteria generally is of order the inverse of the fastest rate in the network (see the discussion in §3.2 and in Ch. 16 of Ref. [5]). For the calculations illustrated in Fig. 6(b), the inverse of the fastest rate gives the lower curve in Fig. 6(c). Thus a normal explicit algorithm would be restricted by stability requirements to timesteps lying approximately in the shaded region below this curve ($dt \simeq 10^{-7}$ seconds or less).

In contrast, Fig. 6(c) displays two curves for stable FLFD integration timesteps lying far above this region. The curve marked dt is for a timestep small enough to give accuracy comparable to Fig. 6(a). This timestep is seen to be about 10^5 times larger than would be stable for a normal explicit integration. The curve marked dt' is for a much larger FLFD algorithm timestep that compromises accuracy for the weaker transitions but remains stable and calculates the stronger transitions correctly. The timestep $dt' \sim 100$ seconds is about

10^9 times larger than would be stable for a standard explicit algorithm. Since dt' already is comparable to the characteristic timescale of a nova explosion, this example exhibits a stable explicit integration timestep for a realistic, extremely stiff system that is effectively arbitrarily large with respect to the usual upper limit for explicit integration.

The picture that emerges from the present results is that the FLFD method is susceptible to stiffness instability, just as for any other explicit method. However, the effect of the instability can be confined to controlled errors in small populations for a range of timesteps far beyond the normal onset of stiffness instability simply by imposing a flux limiter. This flux limiter does not prohibit negative populations but prevents their export between boxes in the network and thus prevents them from growing in uncontrolled fashion.

6. Stiffness under Equilibrium and Non-Equilibrium Conditions

The results of the preceding section indicate that even a relatively crude limit on propagation of negative population can remove large amounts of stiffness and leads to an algorithm useful in realistic applications, even for the extremely stiff networks common in astrophysics. But we now will demonstrate that one can do much better than that, by exploiting the algebraic structure of the differential equations to make a better approximation than just constraining the sign of propagating populations.

The first step is that we must look more deeply at the stiffness instability. As we now discuss, for the sort of large and very stiff networks that we are addressing here there are several fundamentally different sources of stiffness instability that are often not clearly distinguished in the literature. The examples discussed to this point (and in many textbooks) emphasize the type of instability associated with small quantities that should strictly be non-negative becoming negative because of an overly ambitious numerical integration step. The discussion of the flux-limited forward difference algorithm in §4 illustrates that this type of instability can be removed by approximations that do not permit unphysical negative quantities to influence the rest of the network. However, there are other stiffness instabilities that may be initiated even when no population variables become negative in an integration step. In this kind of instability we end up having to take the difference of large numbers to obtain a result very near zero. The numerical errors that ensue in a standard explicit approach can then accumulate rapidly and destabilize the network, even before any abundances become negative. This may still be viewed as a stiffness instability because it results from a numerical integration trying to deal with very different timescales, but the origin of these timescales is different from that discussed above. In this case the disparate timescales are the very rapid reactions driving the system to equilibrium contrasted with the very slow timescale associated with equilibrium itself (which tends to infinity).

As we now consider, this distinction is essential to what follows because these stiffness instabilities have essentially different solutions. Furthermore, we shall find that the second kind of instability can be divided into two subclasses requiring different stabilizing approximations, and that the approximations that we shall introduce in these cases will also take care naturally of the first class of stiffness instabilities because they will as a matter of

course prevent the occurrence of negative probabilities in the network.

6.1. The Approach to Equilibrium

We shall use "equilibrium" in a broad sense to mean a condition where the populations in a network are being strongly influenced by competition between terms of opposite sign on the right sides of the differential equations governing their evolution. How do we measure the degree of equilibration in a large, stiff network? In terms of the coupled set of differential equations describing the network, we may distinguish two qualitatively different conditions:

(i) A macroscopic equilibration that acts at the level of individual differential equations.

(ii) A microscopic equilibration that acts at the level of individual terms within a given differential equation.

Let us consider each of these cases in turn. The differential equations that we must solve take the general form of Eq. (1), $dy_i/dt = F_i^+ - F_i^-$, where the total flux has been decomposed into a component F_i^+ increasing the population y_i and a component F_i^- depleting the population y_i in a given timestep.

6.1.1. Macroscopic Equilibration One class of approximations that we will investigate depends upon assuming that $F_i^+ - F_i^- \to 0$ (asymptotic approximations) or $F_i^+ - F_i^- \to$ constant (steady-state approximations). We shall refer to these conditions as a macroscopic equilibration, since they involve the entire right side of a differential equation in Eq. (1) tending to zero or a finite constant. We shall introduce approximations exploiting this whereby whole differential equations are removed from the numerical integration for a network timestep in favor of a algebraic approximate solutions for that timestep. Such approximations don't reduce the number of equations to integrate, but they reduce the number of equations integrated *numerically by forward difference*. They reduce stiffness for any remaining equations that are integrated numerically in the timestep because removing the equations satisfying these conditions tends to reduce the disparity in timescales for the remaining equations.

6.1.2. Microscopic Equilibration In Eq. (1), F_i^+ and F_i^- for a given species i generally each consist of a number of terms depending on the other populations in the network,

$$\begin{aligned}\frac{dy_i}{dt} &= F_i^+ - F_i^- \\ &= (f_1^+ + f_2^+ + \ldots)_i - (f_1^- + f_2^- + \ldots)_i \\ &= (f_1^+ - f_1^-)_i + (f_2^+ - f_2^-)_i + \ldots = \sum_j (f_j^+ - f_j^-)_i, \end{aligned} \tag{4}$$

At the more microscopic level, groups of individual terms on the right side of Eq. (4) may come approximately into equilibrium (the sum of their fluxes tends to zero), even if macroscopic equilibration conditions are not satisfied. The simplest possibility for this *microscopic equilibration* is that forward–reverse reaction pairs such as $A + B \rightleftharpoons C$, which

will contribute flux terms with opposing signs on the right sides of differential equations in which they participate, come approximately into equilibrium. Then we may consider an algebraic approximation that removes groups of such terms from the numerical integration, replacing their sum of fluxes identically with zero. This will not generally reduce the number of equations to be integrated numerically in a network timestep, but it can reduce dramatically the stiffness of those equations by removing terms with fast rates from the equations, thereby reducing the disparity between the fastest and slowest timescales in the system.

Such considerations will be the basis of the partial equilibrium methods that will be discussed in depth in the third paper in this series [2]. There we shall also demonstrate two important general conclusions: (1) Approximations based on microscopic equilibration are much more efficient at removing stiffness than those based on macroscopic equilibration, because they target more precisely the sources of stiffness in the network. (2) The most powerful approach will be to use macroscopic and microscopic approximations simultaneously in the same set of equations, because they can complement each other in removing stiffness from the equations to be integrated numerically.

6.2. A Quantitative Measure of Microscopic Equilibration

The preceding section suggests that when microscopic equilibration becomes important in a network, the methods for dealing explicitly with stiffness are different from those used to deal with macroscopic equilibration. Therefore, it is important to establish a quantitative measure of how much microscopic equilibration is present in the network. We shall introduce the simplest possibility: that the amount of microscopic equilibration in the network is measured by the fraction of reaction pairs such as $A + B + \ldots \rightleftharpoons C + D + \ldots$ that are judged to be in equilibrium (with each reaction pair considered in isolation from the rest of the network for purposes of this determination).

The full machinery to carry this out will be described in the third paper of this series [2], but our essential approach will be to extend the formalism introduced by Mott [13] to derive conditions on the populations for equilibrium in a reaction pair, and to determine whether the pair is in equilibrium by comparing the actual populations computed from the preceding network timestep with the theoretical equilibrium populations. The general result that we shall obtain is that for a forward–reverse reaction pair of the form $A + B + \ldots \rightleftharpoons C + D + \ldots$, the equilibrium abundance $\bar{y}_i$ of each species is given by

$$\bar{y}_i \equiv y_i^{\mathrm{eq}} = -\frac{1}{2a}(b + \sqrt{-q}). \tag{5}$$

where $q \equiv 4ac - b^2$, the parameters a, b, and c are known functions of the current rate parameters and the populations at the beginning of the timestep, and the approach to equilibrium for the reaction pair is governed by a single timescale $\tau = q^{-1/2}$. We may then estimate whether a given reaction is near equilibrium at time t by requiring

$$\frac{|y_i(t) - \bar{y}_i|}{\bar{y}_i} < \varepsilon_i \tag{6}$$

for each species i involved in the reaction, where $y_i(t)$ is the actual abundance, $\bar{y_i}$ is the equilibrium abundance determined by Eq. (5), and ε_i is a user-specified tolerance that we shall choose typically to be of order 10^{-2}. Alternatively, we may use the equilibration timescale τ compared with the numerical timestep being taken as a measure of microscopic equilibration.

We shall term a network *strongly (microscopically) equilibrated* if a significant fraction of its reaction pairs satisfy the condition (6) or its equilibration timescale τ is much less than than the current integration timestep, and *weakly (microscopically) equilibrated* if at most a few percent of its reaction pairs satisfy Eq. (6) or have values of τ considerably larger than the integration timestep. The remainder of this paper will deal with methods based on asymptotic approximations to stabilize explicit integration for networks that are at most weakly equilibrated, and a second paper will deal with methods based on quasi-steady-state approximations to stabilize weakly-equilibrated systems [1]. The corresponding stabilization of networks near microscopic equilibrium will be discussed in the third paper of this series [2].

7. Algebraic Stabilization of Solutions Using Asymptotic Approximations

The flux-limited forward difference approximation described in §5 illustrates the basic principle that explicit integration in non-equilibrium situations can be stabilized by forbidding the propagation of negative flux. However, the FLFD algorithm represents only a zero-order solution to this problem (a yes/no decision on whether flux components are allowed to propagate) that can be improved substantially by exploiting the structure of the coupled equations to replace the numerical solution with an algebraic approximation that is strictly non-negative. Although the approximations that we now discuss have been implemented in some form in earlier literature [6, 14, 15, 13], we shall find that our implementation appears to be much more successful than previous applications to large, extremely stiff networks, and we shall reach different conclusions about these methods than those reached in earlier publications.

7.1. Some Explicit Asymptotic Approximations

The differential equations that we must solve take the form given by Eq. (1). Generally, F_i^+ and F_i^- for a given species i each consist of a number of terms depending on the other populations in the network. The depletion flux for the population of species i will be proportional to y_i,

$$F_i^- = (k_1^i + k_2^i + \ldots + k_m^i) y_i \equiv k^i y_i, \tag{7}$$

where the k_j^i are rate parameters (in units of time^{-1}) for each of the m processes that can deplete y_i, which may depend on the populations y_j and on hydrodynamical variables such as temperature and density. The characteristic timescales $\tau_j^i = 1/k_j^i$ will differ by many orders of magnitude in the systems of interest, implying that the equations are stiff. From Eq. (7)

we may define the effective total depletion rate k^i for y_i at a given time, and a corresponding timescale τ^i as

$$k^i \equiv \frac{F_i^-}{y_i} \qquad \tau^i = \frac{1}{k^i}, \tag{8}$$

permitting Eq. (1) to be written as

$$y_i = \frac{1}{k^i}\left(F_i^+ - \frac{dy_i}{dt}\right). \tag{9}$$

Thus, in a finite-difference approximation at timestep t_n we have

$$y_i(t_n) = \frac{F_i^+(t_n)}{k^i(t_n)} - \frac{1}{k^i(t_n)}\left.\frac{dy_i}{dt}\right|_{t=t_n}. \tag{10}$$

We now define the *asymptotic limit* for the species i to be $F_i^+ \simeq F_i^-$, implying from Eq. (1) that $dy_i/dt \simeq 0$. In this limit Eq. (10) gives a first approximation $y_i^{(1)}(t_n)$ and local error $E_n^{(1)}$, respectively, for $y_i(t_n)$ as

$$y_i^{(1)}(t_n) = \frac{F_i^+(t_n)}{k^i(t_n)} \qquad E_n^{(1)} \equiv y(t_n) - y^{(1)}(t_n) = -\frac{1}{k(t_n)}\frac{dy}{dt}(t_n) \tag{11}$$

For small dy_i/dt we may then get a correction term by writing the derivative term in Eq. (10) as

$$\frac{dy}{dt}(t_n) = \frac{1}{\Delta t}\left(y_i(t_n) - y_i(t_{n-1})\right) + \frac{1}{\Delta t}\left(E_n^{(1)} - E_{n-1}^{(1)}\right) + O(\Delta t), \tag{12}$$

where $O(x)$ denotes order of x. If we retain only the first term and approximate $y(t)$ by $y^{(1)}(t)$, we obtain

$$\left.\frac{dy}{dt}\right|_{t=t_n} \simeq \frac{1}{\Delta t}\left(y_i^{(1)}(t_n) - y_i^{(1)}(t_{n-1})\right) = \frac{1}{\Delta t}\left(\frac{F_i^+(t_n)}{k^i(t_n)} - \frac{F_i^+(t_{n-1})}{k^i(t_{n-1})}\right). \tag{13}$$

Therefore, the estimate for y is improved to

$$y_n^{(2)} \simeq \frac{F_n^+}{k_n} - \frac{1}{k_n \Delta t}\left(\frac{F_n^+}{k_n} - \frac{F_{n-1}^+}{k_{n-1}}\right), \tag{14}$$

where we now employ compact index notation $y_n \equiv y_i(t_n)$, and so on, and have dropped the species index i to avoid notational clutter. Because we are approximating the derivative term, we expect that Eq. (14) is valid only if the second term is small, implying that our approximation becomes more valid if $k\Delta t$ is large.

The preceding discussion implements an asymptotic approximation in a very simple way. Other more sophisticated asymptotic approximations may be derived. For example, if we retain the full expression for the derivative in Eq. (12) and substitute in Eq. (10), we obtain

$$y_n^{(2)} = \frac{F_n^+}{k_n} - \frac{1}{k(t_n)\Delta t}\left(y(t_n) - y(t_{n-1}) - \frac{1}{k(t_n)\Delta t}\left(-E_n^{(1)} + E_{n-1}^{(1)}\right).\right.$$

Setting $y(t_n) = y_n^{(2)}$ and solving for $y_n^{(2)}$ gives

$$y_n^{(2)} = \frac{1}{1 + k_n \Delta t}\left(y_{n-1} + F_n^+ \Delta t\right), \tag{15}$$

where a term of order $(\Delta t)^2/(1+k(t_n)\Delta t)$ has been discarded. Another approach is to use a predictor–corrector scheme [6, 15, 13]. We may solve Eq. (1) by finite difference, replacing quantities on the right side of Eq. (1) with averaged quantities:

$$\frac{y_n - y_{n-1}}{\Delta t} = \tfrac{1}{2}(F_n^+ + F_{n-1}^+) - \frac{y_n + y_{n-1}}{\tau_n + \tau_{n-1}}, \tag{16}$$

where for notational convenience we have traded the rate parameters k_i for the corresponding timescales $\tau_i \equiv 1/k_i$. Solving this equation for y_n gives

$$y_n = \frac{(\tau_n + \tau_{n-1} - \Delta t)y_{n-1} + \frac{1}{2}(\tau_n + \tau_{n-1})(F_n^+ + F_{n-1}^+)\Delta t}{\tau_n + \tau_{n-1} + \Delta t}, \tag{17}$$

which is not easy to use because the τ_n and F_n^+ are implicit functions of the y_n that we seek. We obtain a usable explicit algorithm by solving an approximate form of Eq. (17) to get a first guess $y^{\rm p}$ for y_n (the predictor step), using that approximate result to estimate values of τ_n and F_n^+, and then using these values obtained from the predictor step to solve Eq. (17) (the corrector step). Specifically,

- The predictor estimate $y^{\rm p}$ is obtained by setting $\tau_n = \tau_{n-1} \equiv \tau_0$ and $F_n^+ = F_{n-1}^+ \equiv F_0^+$ in Eq. (17).
- The corrector value $y^{\rm c}$ is obtained by substituting $F_{\rm p}^+$ for F_n^+, and $\tau_{\rm p}$ for τ_n in Eq. (17).

This yields a predictor and corrector [6]

$$y^{\rm p} = \frac{y^0(2\tau_0 - \Delta t) + 2F_0^+\tau_0\Delta t}{2\tau_0 + \Delta t} \qquad \text{(Predictor)}$$
$$y^{\rm c} = \frac{y^0(\tau_{\rm p} + \tau_0 - \Delta t) + \frac{1}{2}\Delta t(F_{\rm p}^+ + F_0^+)(\tau_{\rm p} + \tau_0)}{\tau_{\rm p} + \tau_0 + \Delta t} \qquad \text{(Corrector)} \tag{18}$$

where "0" denotes initial quantities and "p" denotes quantities computed using the results of the predictor step. We shall test the asymptotic approximations (14), (15), and (18) in examples below.

7.2. Mathematical Properties of Asymptotic Approximations

The mathematical and numerical properties of asymptotic approximations have been explored in Refs. [6, 14, 15, 13] and we only summarize them here.

(i) The preceding asymptotic formulas are explicit, since only quantities already known are required to advance a timestep.

(ii) Asymptotic methods do not need to compute Jacobians or invert matrices.

(iii) Asymptotic methods are A-stable [6, 16] on linear problems.

(iv) Explicit asymptotic methods should scale linearly with network size.

(v) Asymptotic methods require initial values from only one timestep (self-starting).

A more extensive discussion may be found in Oran and Boris [6] and in Mott [13].

7.3. An Asymptotic Flux-Limiting Algorithm

We now use the preceding formalism to define an explicit asymptotic flux-limiting integration algorithm. Since the asymptotic approximation specified above is expected to be valid if $k\Delta t$ is large, we define a critical value κ of $k\Delta t$ and at each timestep cycle through all network populations and compute the product $k^i\Delta t$ for each species i using Eq. (8) and the proposed timestep Δt. Then, for each population species i

(i) If $k^i\Delta t < \kappa$, we update the population numerically using the standard flux-limiting explicit algorithm discussed in §4.

(ii) Otherwise, for $k\Delta t \geq \kappa$, we update the population algebraically using one of the asymptotic approximations given in Eqs. (14), (15), or (18).

From considerations such as those in §3 we expect an explicit integration to be stable if $k^i\Delta t < 1$ and to potentially be unstable if $k^i\Delta t \geq 1$, a possible choice is $\kappa = 1$. For the networks discussed in this paper, we have found this value of κ to work well and have adopted it. For that specific choice of κ, explicit numerical integration should be stable for those cases where it is applied, so we expect that at each timestep all abundances will remain positive and the flux-limiting prescription for the explicit numerical integration may be dropped. Notice that at each timestep some species may be updated by explicit forward difference and some by the asymptotic approximation, and that the division of species between these two categories could change at each timestep since the product $k^i\Delta t$ is time-dependent.

7.4. A Simple Adaptive Timestepper

Implementing the preceding algorithm requires an adaptive timestepper. We take the point of view that the present task is to establish whether explicit methods can even compete with implicit methods for stiff networks. Since previous studies have generally found that explicit methods fail by many orders of magnitude to attain the speeds of implicit methods in highly-stiff networks, our timestepper need not be highly optimized at this point to answer that question.

The first consideration is a standard one that limits the population change at each timestep. However, the preceding algorithm does not guarantee that total population is conserved, so in setting an adaptive timestep one should make a check that ensures conservation of population at the desired level. Generally, if population conservation does not satisfy the required tolerance at a given timestep, making the timestep small enough is guaranteed to improve conservation of population because it will reduce $k^i\Delta t$ and therefore will tend to decrease the number of isotopes being treated asymptotically. To ensure conservation of particle number at a desired level in the overall calculation, we may limit the deviation in any one timestep to a small amount. Thus, we adopt a simple timestepper with two stages:

(i) After the rates and fluxes are computed at the beginning of a new timestep, compute a trial timestep based on limiting the change in populations that would result from that timestep to some specified tolerance. Use the minimum of this trial timestep and the

Table 1. Explicit speedup factors F

Network	Isotopes	Speedup F
pp	6	~ 1.5
Alpha	16	3
Nova	134	7
150-isotope	150	7.5
365-isotope	365	~ 20

timestep that was taken in the previous integration timestep to update the populations by the explicit asymptotic algorithm.

(ii) Check for conservation of population. If the conservation law is satisfied within the desired tolerance for this timestep, proceed. If it is not, or is satisfied too well, decrease or increase the timestep as appropriate by a small factor and repeat the calculation of populations with the new timestep but original fluxes.

We then accept this timestep, without further check. One could also iterate to ensure that the conservation condition is satisfied at each step, but this did not significantly improve the results in our tests.

Far-removed from equilibrium the limitation in population changes determines the timestep with this algorithm, but in the approach to equilibrium the timestep becomes dominated by the probability conservation criterion. Though it is likely not very optimized, we have found this simple timestepper to be stable and accurate for the varied astrophysical thermonuclear networks that we have tested, and thus adequate for our task here. This should be contrasted with previous attempts to apply asymptotic methods to thermonuclear networks, which failed to produce accurate results and were abandoned as unsuitable for such stiff networks [6, 13].

8. Comparisons of Explicit and Implicit Integration Speeds

We shall be comparing explicit and implicit methods using codes that are at very different stages of development and optimization. We assume that for codes at similar levels of optimization the primary difference between explicit and implicit methods would be in the extra time spent in implicit-method matrix operations. Hence, if the fraction of time spent on linear algebra is f for an implicit code, we assume that an explicit code at a similar level of optimization could compute a timestep a factor of $F = 1/(1-f)$ faster. In Table 1 we display factors F based on data obtained by Feger [17, 18] using the implicit, backward-Euler code Xnet [11] with both dense and sparse solvers. Then we may compare roughly the speed of explicit versus implicit codes (possibly at different levels of optimization) by multiplying F by the ratio of implicit to explicit integration steps required for a given problem. This procedure has obvious uncertainties, and likely underestimates the speed of an optimized explicit versus optimized implicit code [2], but will give a useful lower limit on how fast the explicit calculation can be.

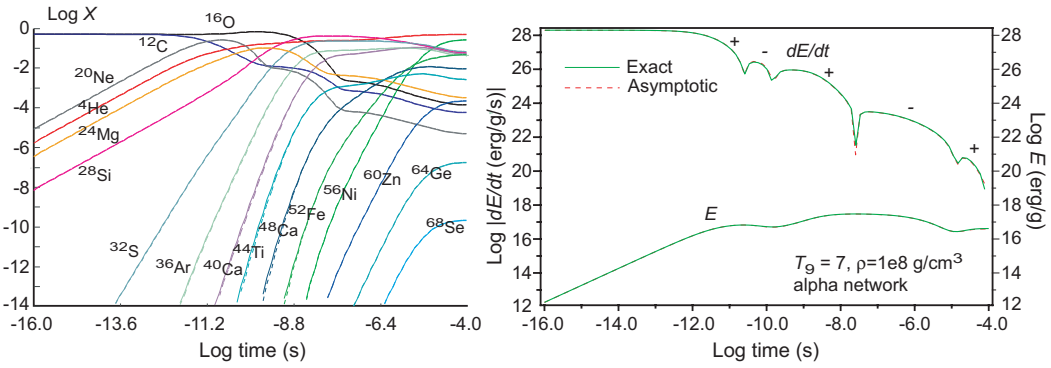

Figure 7. Comparison of the asymptotic approximation (14) with a forward Euler integration that used timesteps short enough to be stable for an alpha network at constant temperature $T_9 = 7$ and density of 10^8 g cm^{-3}. Initial abundances corresponded to equal mass fractions of ^{12}C and ^{16}O, and rates from the REACLIB library [10] were used. The left figure compares mass fractions; the right figure compares differential and integrated energy production. Solid lines are explicit forward-Euler integration with a timestep constrained to be less than the inverse of the fastest rate in the system so that it is stable; dashed lines are the corresponding asymptotic approximation using Eq. (14). The value of dE/dt oscillates between positive and negative values so the log of the absolute value of dE/dt is plotted, with the sign of dE/dt indicated in each region.

9. Network Calculations in Simplest Asymptotic Approximation

First we shall establish that the asymptotic algorithm is capable of correct integration of coupled sets of extremely stiff equations. We have tested this for a variety of calculations in two ways:

(i) Comparisons with results from standard implicit codes, and

(ii) For some smaller networks where the corresponding integration time is not prohibitive, comparison with explicit forward Euler calculations made with timesteps short enough to be stable.

Our general finding is that the asymptotic algorithm outlined above gives the same results as standard implicit and explicit codes, even for the stiffest networks found in astrophysics applications, provided that the numerical timesteps are limited sufficiently to ensure conservation of overall probability in the network at the desired level of precision.

9.1. Tests Against Fully-Explicit Calculations

Figure 7 illustrates an asymptotic approximation calculation that gives results essentially identical to results from exact numerical integration. In this example the sum of the mass fractions was constrained to deviate from unity by not more than 1% over the entire range of the calculation by requiring that if it deviates by more than 10^{-8} from unity in any one timestep, the timestep is reduced in size. We shall generally use this global 1% criterion for examples presented in this paper. Higher precision may be obtained by tightening

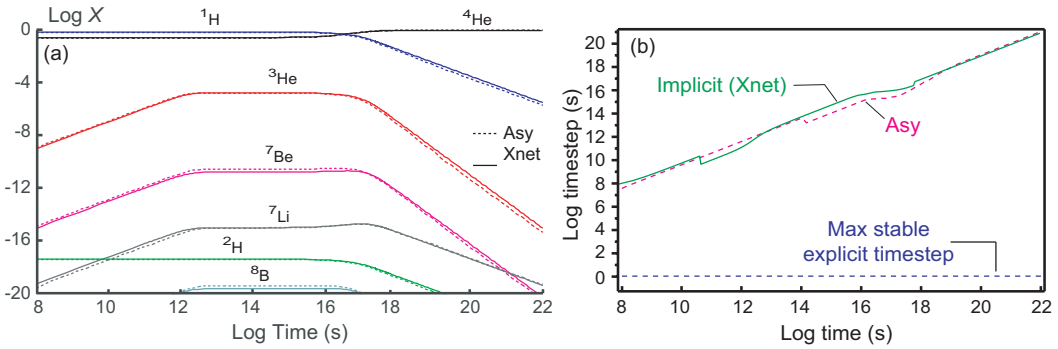

Figure 8. Integration of the pp-chains under constant temperature and density characteristic of the core of the present Sun: $T_9 = 0.016$ and a density of 160 g cm^{-3}, assuming solar initial abundances. Reaction rates were taken from the REACLIB library [10]. (a) Mass fractions for the asymptotic method of Eq. (14) (dotted curves) and for the standard implicit code Xnet [11] (solid curves). (b) Integration timesteps for the asymptotic method (dotted magenta) and the implicit method (solid green). The expected maximum stable fully-explicit timestep is indicated by the dashed blue curve.

this constraint, but this is typically already more conservative than justified since input parameter uncertainties in realistic large thermonuclear networks and uncertainties in the coupled hydrodynamics may each be considerably larger than 1%.

Having established that explicit asymptotic approximations can give correct results even for extremely stiff networks, we now turn to the question of their efficiency (and further tests of accuracy) by examining calculations of various astrophysical thermonuclear networks using this approximation. In this section we shall use the simplest asymptotic approximation, corresponding to Eq. (14). In §10 we shall test the alternative asymptotic approximations of Eqs. (15) and (18).

9.2. Explicit Asymptotic Integration of the pp-Chains

The solar pp-chains provide a striking example of stiffness in a simple network. In Fig. 8 we illustrate integration of the pp-chains at a constant temperature and density characteristic of the core in the present Sun, using the asymptotic method and the implicit backward-Euler code Xnet [11]. We see that the explicit asymptotic integration gives results for the mass fractions in rather good agreement with the implicit code over 20 orders of magnitude, and is generally taking timesteps $dt \sim 0.1t$ that are comparable to those for the implicit code over the entire range of integration. (The asymptotic method required 333 total integration steps versus 176 steps for the implicit code.)

As shown in §3, the maximum stable timestep for a standard explicit integration method may be estimated as the inverse of the fastest rate contributing to the network. This is illustrated by the dashed blue curve in Fig. 8(b). At late times the explicit integration timesteps are $\sim 10^{21}$ times larger than the maximum stable timestep for a normal explicit integration. The calculation illustrated in Fig. 8 takes less than a second on a 3 GHz processor with the explicit asymptotic method (as does the implicit solver). In contrast, from Fig. 8(b) we

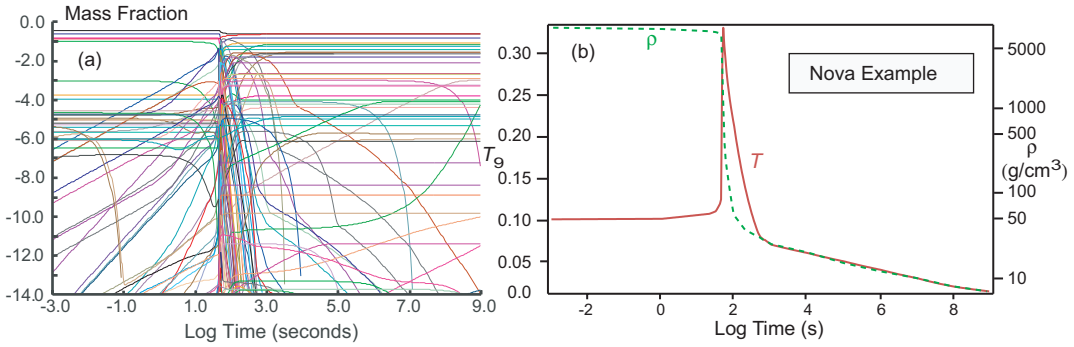

Figure 9. (a) Mass fractions for a network under nova conditions, corresponding to the hydrodynamical profile shown in (b). The calculation used the explicit asymptotic method corresponding to Eq. (14) and a network containing 134 isotopes coupled by 1531 reactions, with rates taken from the REACLIB library [10] and initial abundances enriched in heavy elements [12].

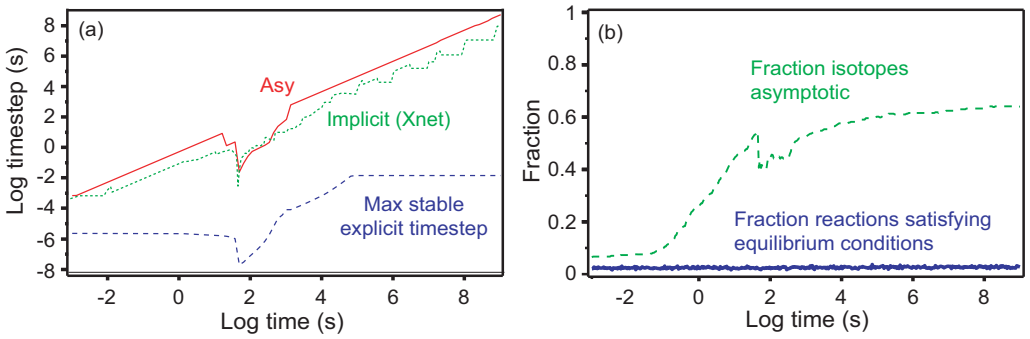

Figure 10. (a) Timesteps for integration of Fig. 9. The solid red curve is from the asymptotic calculation. The dotted green curve is from an implicit integration using the backward-Euler code Xnet [11]. The dashed blue curve estimates the largest stable fully explicit timestep as the inverse of the fastest rate in the system. (b) Fraction of isotopes that become asymptotic and fraction of reactions that reach partial equilibrium in the asymptotic-method calculation.

estimate that a standard explicit method taking the largest stable fully-explicit timestep would require a time about 1000 times longer than the age of the Universe ($\sim 10^{21}$ s of processor time) to compute the pp-chains to hydrogen depletion.

9.3. Simulations under Nova Conditions

The preceding example entailed an exceedingly stiff but rather small network. Let us now turn to an example that is highly stiff and involves hundreds of isotopes. In Fig. 9(a) we illustrate a calculation using the explicit asymptotic algorithm and a hydrodynamical profile shown in Fig. 9(b) that is characteristic of hot-CNO burning in a nova outburst. The explicit asymptotic timesteps are displayed in Fig. 10(a). In these simulations we see that the asymptotic network takes stable and accurate timesteps corresponding to $dt \sim 0.1t$ over most of the time integration, except in the region of sharp temperature rise and strong burning,

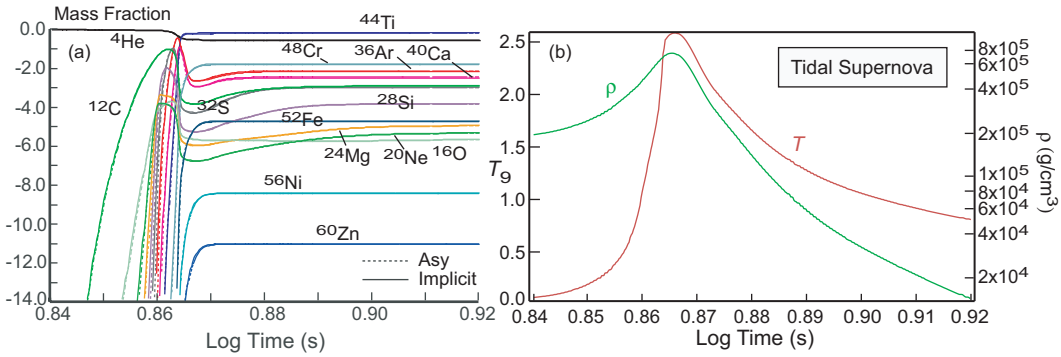

Figure 11. (a) Mass fractions for an alpha network under tidal supernova conditions with an initial abundance of pure ^{4}He. The network contained 16 isotopes coupled by 46 reactions, with rates from REACLIB [10]. Dotted curves calculated in asymptotic approximation using Eq. (14). Solid curves calculated using the fully implicit code Xnet [11]. (b) The corresponding hydrodynamical profile [19].

where $dt \sim (0.01 - 0.001)t$. Over most of the range of integration after burning commences, the asymptotic solver timesteps (solid red curve in Fig. 10(a)) are a million or more times larger than the maximum stable timestep for a purely explicit method (dashed blue curve in Fig. 10(a)); at late times this disparity increases and by the end of the calculation the asymptotic timesteps are approximately 10^{10} times larger than would be possible for a normal explicit integration.

The generally large explicit asymptotic timesteps over the entire integration range illustrated in Fig. 10(a) are greater than or equal to those for a typical implicit code, as may be seen by comparing with the implicit (backward Euler) calculation timestepping curve shown in dotted green. In this calculation the implicit method required 1335 integration steps while the explicit asymptotic calculation required only 935 steps, and furthermore we expect that the explicit timesteps can be computed more quickly than the implicit timesteps. For a network with 134 isotopes, an optimized explicit code should calculate a timestep perhaps 7 times faster than typical implicit codes (Table 1). These large explicit timesteps are possible because during the simulation many isotopes become asymptotic but few reactions reach partial equilibrium, as illustrated in Fig. 10(b). Similar results for nova simulations were obtained with the asymptotic method in Refs. [17, 18] using a different nova hydrodynamical profile and a different reaction library. We conclude that the explicit asymptotic method may intrinsically be an order of magnitude faster than a state-of-the art implicit code for simulations under nova conditions.

9.4. Simulations under Tidal Supernova Conditions

The mass fractions as a function of time for a thermonuclear supernova event induced by tidal interaction in a white-dwarf [19] are illustrated in Fig. 11(a) for an alpha network and the hydrodynamical profile is illustrated in Fig. 11(b). Dotted curves correspond to the explicit asymptotic calculation and solid curves correspond to a fully-implicit, backward-

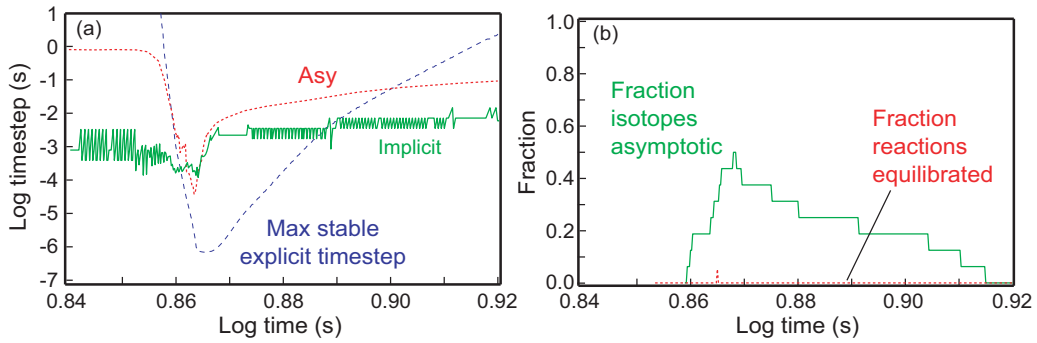

Figure 12. (a) Asymptotic integration timesteps (dotted red), integration steps for the implicit code Xnet [11] (solid green) and maximum stable purely explicit step (dashed blue) for the calculation in Fig. 11. (b) Fraction of isotopes that become asymptotic and fraction of reactions equilibrated in the network.

Euler calculation using Xnet [11]. The mass fractions for the two calculations are almost indistinguishable. The timestepping is compared for the asymptotic calculation and Xnet in Fig. 12(a). We see that the timestepping for the asymptotic calculation is somewhat better than for the implicit code (480 total integration steps for the asymptotic calculation versus 2136 total integration steps for the implicit calculation). Estimating that a fully-optimized explicit solver can calculate a timestep about 3 times faster than an implicit code like Xnet in a 16-isotope network (Table 1), we surmise that the asymptotic method is capable of doing the integration for Fig. 12 perhaps 10 times faster than a current implicit code. The relatively good timestepping for the asymptotic method in this case is because essentially no reactions in the network come into equilibrium, as illustrated in Fig. 12(b). The flat mass fraction curves at late times in Fig. 11(a) are not a result of equilibrium but rather of reaction freezeout caused by the temperature and density dropping quickly at late times as the system expands (Fig. 11(b)).

A calculation for the hydrodynamical profile illustrated in Fig. 11(b) but for a 150-isotope network is illustrated in Fig. 13. We note that in this example the asymptotic approximation permits timesteps that are typically 8–9 orders of magnitude larger than the maximum stable explicit timestep (indicated by the dashed blue line). These timesteps are again competitive with those of a standard implicit code (shown as the solid green curve in Fig. 13(b)). In this case the implicit code took 2425 timesteps to complete the integration while the asymptotic method took 5593 integration steps, but this factor of about two fewer implicit timesteps would be more than offset by the significantly faster computation of each integration step expected for an optimized explicit asymptotic code integrating a network this large. For a 150-isotope network we expect that an optimized explicit code would be more than 7 times faster computing a timestep than an implicit code (Table 1), so a fully-optimized asymptotic method is probably capable of doing the integration in Fig. 13 several times faster than a current implicit code.

A calculation for the hydrodynamical profile illustrated in Fig. 11(b) for a 365-isotope network is illustrated in Fig. 14. We note that in this example the asymptotic approximation

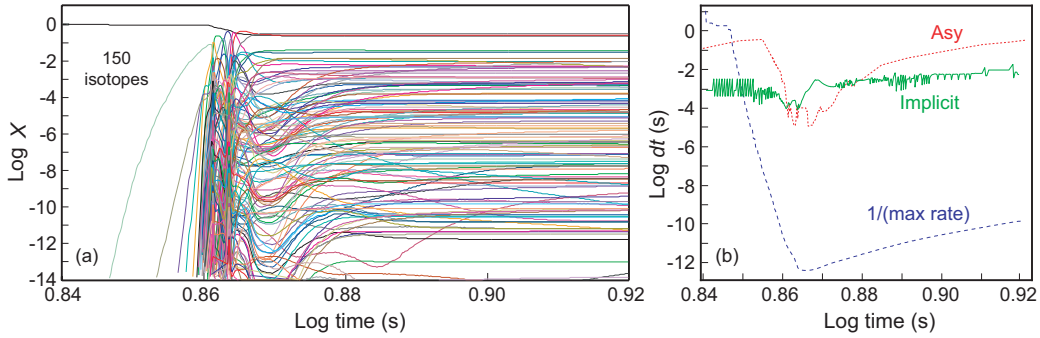

Figure 13. (a) Mass fractions for a 150-element network with 1563 reaction couplings under tidal supernova conditions, corresponding to the hydrodynamical profile shown in Fig. 11(b). Initially the abundance was pure ^{4}He and rates were taken from REACLIB [10]. The mass fractions were calculated in asymptotic approximation using Eq. (14). (b) The corresponding integration timesteps for the asymptotic method (dashed red) and the implicit code Xnet [11] (solid green). The maximum stable explicit timestep is estimated by the dashed blue curve.

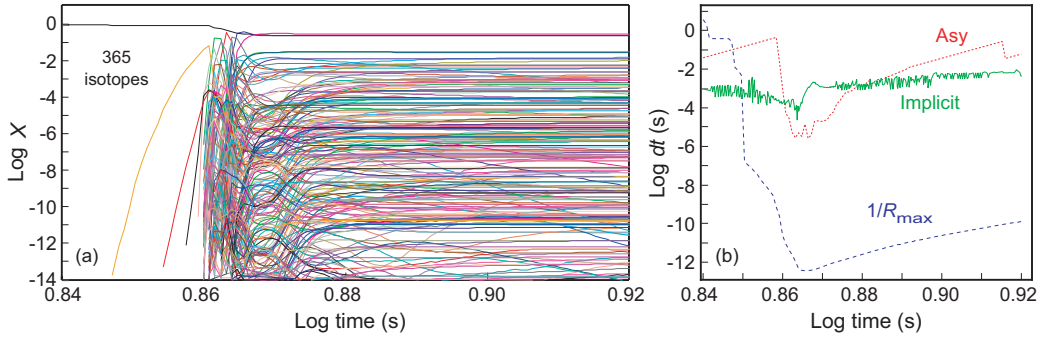

Figure 14. (a) Mass fractions calculated in asymptotic approximation using Eq. (14) for a 365-element network with 4325 reaction couplings under tidal supernova conditions, corresponding to the hydrodynamical profile shown in Fig. 11(b). Initially the abundance was pure ^{4}He and rates were taken from REACLIB [10]. (b) The corresponding integration timesteps for the asymptotic method (dashed red curve) and the implicit code Xnet [11] (solid green curve). The maximum stable explicit timestep is indicated by the dashed blue curve.

permits timesteps that are as much as 10^{10} times larger than the maximum stable explicit timestep (dotted blue curve). This timestep is again competitive with that of an implicit calculation (solid green curve in Fig. 14(b)). The implicit calculation required 2707 integration steps, compared with 5778 steps for the asymptotic calculation. But as noted above, this factor of two advantage of the implicit calculation should be more than offset by the much more efficient computation of each timestep in an optimized asymptotic code. For this 365-isotope network we may assume from Table 1 that the explicit code can calculate each timestep ~ 20 times faster than the implicit code, so an optimized asymptotic code should be capable of performing the integration in Fig. 14 perhaps 10 times faster than a state-of-the-art implicit code.

Similar results for networks under tidal supernova conditions have been found in

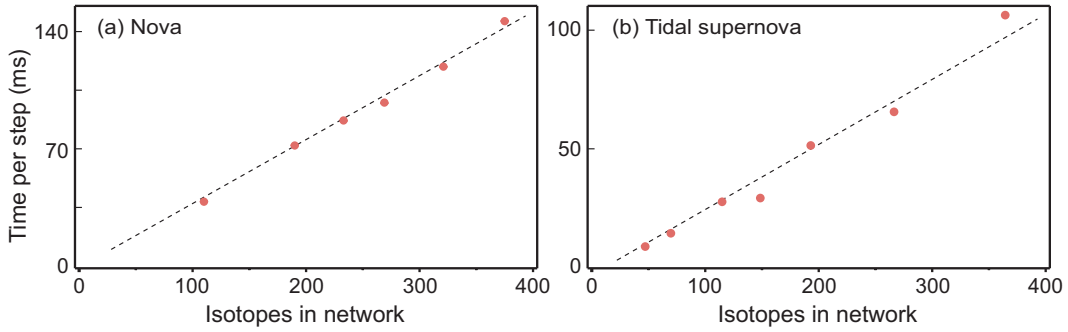

Figure 15. Linear scaling of wall clock time per integration step with number of isotopes in the network for the explicit asymptotic approximation. (a) The nova simulation of Fig. 9. (b) The tidal supernova simulation of Figs. 11–14. The dashed lines are drawn only to guide the eye.

Refs. [17, 18]. Although a different set of reaction network rates was used in these references, the explicit asymptotic method was again found to be highly competitive with standard implicit methods for the tidal supernova problem.

9.5. Scaling with Network Size

In Fig. 15 we illustrate scaling of integration time with network size for the nova simulation of Fig. 9 and the tidal supernova simulation of Figs. 11–14. The behavior is seen to be approximately linear, as expected for an explicit algorithm since no matrix inversions are required.

10. Tests of More Sophisticated Asymptotic Algorithms

All results presented to this point have used the simplest asymptotic formula defined in Eq. (14). This section compares results from Eq. (14) to those obtained with the more sophisticated formulas (15), and (18), for the case of an alpha network with constant $T_9 = 5$ and $\rho = 1 \times 10^8$ g cm^{-3}. In Fig. 16 we display a composite of the computed mass fractions and the timestepping for the asymptotic approximations corresponding to these three formulas. We find that the more sophisticated asymptotic approximations in Eqs. (15) and (18) give results for abundances that are very similar to those from Eq. (14), but can in some cases give more favorable timestepping by factors of several. These results are representative of tests on a variety of networks and we conclude that for typical thermonuclear networks Eqs. (14), (15), and (18) yield similar results, except for possible differences by factors of up to 2–3 in computational speed.

11. Non-Competitive Asymptotic Timesteps in the Approach to Equilibrium

In previous sections evidence has been presented that, well-removed from equilibrium, asymptotic methods can provide stable and accurate integration of the stiffest large networks

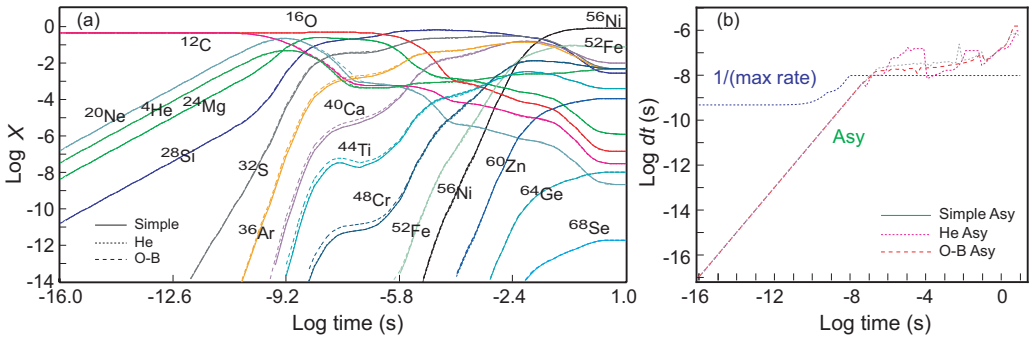

Figure 16. Comparison of mass fractions (a) and timesteps (b) using different asymptotic approximations for an alpha network with constant $T_9 = 5$ and $\rho = 1 \times 10^8$ g cm^{-3}. The network contained 16 isotopes coupled by 46 reactions, with rates taken from REACLIB [10]. Initial abundances corresponded to equal mass fractions of ^{12}C and ^{16}O. Simple refers to Eq. (14), He to Eq. (15), and O–B to Eq. (18).

with timesteps that are comparable to those employed in standard implicit and semi-implicit stiff solvers. In practice, for astrophysical thermonuclear networks this means that timesteps are typically from 0.1 to 0.001 of the current time over most of the integration range, except for brief time periods where very strong fluxes are being produced and timesteps may need to be shorter to maintain accuracy. Since explicit methods can generally compute each timestep substantially faster than for implicit methods, this suggests that such methods offer a viable alternative to implicit solvers under those conditions.

However, the preceding statements are no longer true when substantial numbers of reaction pairs in the network begin to satisfy microscopic equilibrium conditions (according to the criteria given in §6.2). Then the typical behavior for asymptotic approximations is for the timestep to become constant or only slowly increasing with integration time. (Indeed, we have already seen an indication of this behavior at late times in Fig. 16(b)). Figure 17 illustrates for two different networks coupled to a single-zone hydrodynamical simulation. We see that the 19-isotope network is able only marginally to keep up with the hydrodynamical timesteps, while the 150-isotope network lags orders of magnitude behind for hydrodynamical integration times later than about $10^{-4} - 10^{-5}$ s.

The reason for this loss of timestepping efficiency for the asymptotic method is the approach to equilibrium at late times; this is documented in the inset plots for Fig. 17, which show the fraction of reactions in the respective networks that satisfy equilibrium conditions. The reason that the 19-isotope network is able to keep up much better than the 150-isotope network is also clear from the inset plots: the 19-isotope network slowly approaches 30–40% equilibration in this region, but the 150-isotope network, with much faster reactions because it includes fast proton and neutron reactions not found in the 19-isotope network, quickly reaches 70% equilibration. Since in most applications for extremely stiff astrophysical networks the physical phenomena require integration over many decades of time, this lag of the asymptotic timestepping as equilibrium is approached is disastrous for such approximations and they quickly lose out to implicit methods, which can continue to

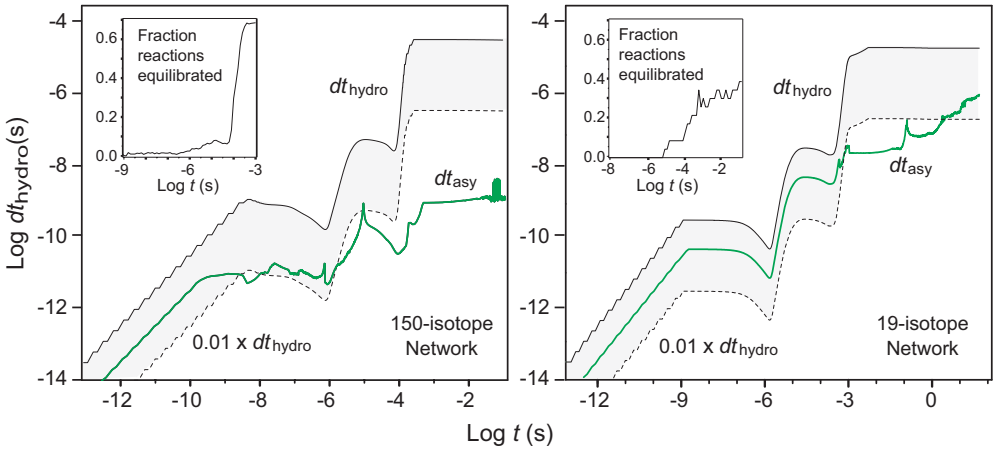

Figure 17. Network timesteps (solid green curves) in asymptotic approximation for a single-zone 2-dimensional hydrodynamics simulation using the Flash [9] hydrodynamics code. Left: 150-isotope network. Right: 19-isotope network. Insets show the fraction of reactions that reach equilibration as a function of time. The shaded regions indicate the target network timestepping required for the network to keep up with the hydrodynamics, as discussed further in conjunction with Fig. 1.

take large integration steps even nearing equilibrium (though they are inefficient at computing each timestep).

Another example of the failure of the asymptotic approximation to generate competitive timesteps is illustrated in Fig. 18. The asymptotic mass fractions calculated in Fig. 18(a) are quite accurate in comparison with the results from standard implicit codes, but the timestepping illustrated in Fig. 18(b) is not competitive with implicit methods at late times. For $\log t \sim -2$, the asymptotic integration timesteps are only of order 10^{-8} seconds, whereas the implicit code is taking timesteps larger than 10^{-3} seconds at this point.

The reason for this failure of the asymptotic approximation can be seen clearly in Fig. 18(b) and (d). In the region of rapid temperature rise illustrated in Fig. 18(c), the rates in the network increase dramatically and this causes the maximum stable explicit timestep to become much smaller, as illustrated in Fig. 18(b) by the dashed blue curve. The steadily increasing explicit timestep (solid green curve) intersects this maximum stable explicit timestep curve near $\log t = -8$. But from Fig. 18(d) we see that essentially no isotopes in the network satisfy the asymptotic condition at this point. Thus the integration is forced to use a standard explicit method and the timestep must (to maintain stability) follow the decreasing dashed blue curve until around $\log t = -4$, when significant numbers of isotopes finally begin to satisfy the asymptotic condition and the explicit asymptotic algorithm is able begin to take timesteps larger than the explicit limit. However, at this point the asymptotic timesteps are already orders of magnitude smaller than those an implicit method would use, and the asymptotic method is able to increase the timestep by only about two orders of magnitude before the system reaches equilibrium. Thus, for the entire time range from $\log t \sim -8$ until $\log t \sim -2$ the explicit asymptotic method computes the network accurately but its timesteps lag many orders of magnitude behind those from standard implicit methods.

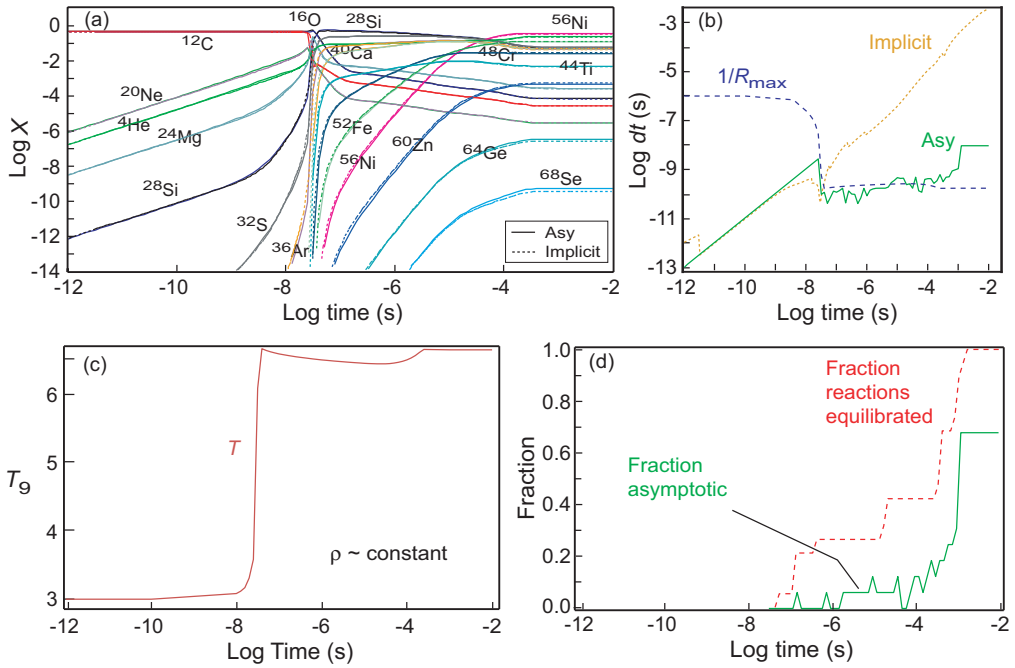

Figure 18. Asymptotic calculation with an alpha network for a hydrodynamic profile characteristic of a single isolated zone in a Type Ia supernova explosion. (a) Mass fractions in asymptotic approximation (solid) and with the implicit code Xnet [11] (dotted). (b) Integration timesteps. The asymptotic timesteps are illustrated by the solid green curve and the Xnet implicit timesteps by the dotted orange curve. The dashed blue curve labeled $1/R_{\rm max}$ indicates approximately the maximum stable timestep for a purely explicit method (corresponding to the inverse of the fastest rate in the network). (c) The hydrodynamic profile. (d) Fraction of isotopes that become asymptotic and the fraction of network reactions that become equilibrated in the course of the explosion.

As we shall explain in considerable detail in the third paper of this series [2], the reason for the loss of efficiency for asymptotic methods as equilibrium is approached is that the asymptotic approximation removes a large amount of stiffness associated with macroscopic equilibration, but near (microscopic) equilibrium a fundamentally new source of stiffness begins to play a role and it is not generally removed by the asymptotic approximation. Indeed, we see clearly from Fig. 18(d) that the fraction of reactions in the network that satisfy equilibrium conditions increases rapidly beginning at $\log t \sim -7$ and reaches unity by $\log t \sim -3$. In this third paper we shall describe a new implementation of *partial equilibrium methods* that can be used in conjunction with asymptotic methods to turn equilibrium from a liability into an asset and increase the explicit timestepping by orders of magnitude in the approach to equilibrium. In that paper we will give examples suggesting that these methods are capable of giving timestepping competitive with that of implicit methods across the entire range of interesting physical integration times for a variety of extremely stiff reaction networks.

12. Summary and Conclusions

Explicit numerical integration can compute a timestep faster then implicit methods, and the time to compute a network explicitly scales linearly and therefore more favorably with network size than for implicit codes. Nevertheless, previous discussions of numerical integration for very stiff systems have concluded rather uniformly that explicit methods are not competitive with implicit methods for stiff networks because they are unable to take large enough stable timesteps. To quote *Numerical Recipes* [5], "For stiff problems we *must* use an implicit method if we want to avoid having tiny stepsizes."

Improvements in explicit methods based on using asymptotic and steady-state limiting solutions to remove stiffness from the network have had some success for systems of moderate stiffness such as various chemical kinetics problems. However, it has been concluded in the previous literature that such methods are not competitive, failing even to give correct results, with timesteps that are far too short to be useful even if they gave correct results, for extremely stiff networks such as those encountered commonly in astrophysical thermonuclear networks [6, 13]. This paper has presented evidence strongly challenging all of these conclusions.

We have cleanly identified three fundamentally different sources of stiffness in large networks, only the first of which is commonly emphasized in the literature:

(i) Situations where small populations can become negative if the explicit timestep is too large, with the propagation of this anomalous negative population leading to exponentially growing terms that destabilize the network.

(ii) Situations where the right sides of the differential equations expressed as $dY = F = F^+ - F^-$ approach a constant derived from the difference of two large numbers (the total flux in F^+ and total flux out F^-), and numerical errors in taking this difference destabilize the network if the timestep is too large.

(iii) Situations where on the right sides of the differential equations expressed in the form of Eq. (4) the net flux in specific forward-reverse reaction pairs $(f_i^+ - f_i^-)$ tends to zero as the system approaches equilibrium, leading to large errors if the timestep is too large because the net flux is derived from the difference of two large numbers and the timescale equilibrating the populations is short compared with the desired numerical timestep.

We have shown that these distinctions are important because different sources of stiffness require different approximations for their removal in an algebraically-stabilized explicit integration.

Using the extremely stiff systems characteristic of astrophysical thermonuclear networks as a stringent test, we have shown that asymptotic methods are very successful at removing the first two types of stiffness, and give correct results, even for the stiffest of thermonuclear networks, provided that adequate attention is paid to conservation of probability in the network. Furthermore, we have shown various examples of stable and accurate timestepping with these methods in extremely stiff systems that are competitive with that of standard implicit codes, demonstrating in some simple but physically-important networks timesteps that are as much as 20 orders of magnitude larger than the maximum timestep that would be

stable in a standard explicit method.

Asymptotic methods are adept at removing the first two types of stiffness listed above, permitting explicit numerical timesteps that are competitive with implicit methods even in the stiffest networks. However, we have also shown that such methods give correct results but fail to exhibit competitive timestepping when the system approaches microscopic equilibrium and the third type of stiffness instability begins to dominate. In a following paper [2], we shall provide evidence for competitive timestepping, even in the approach to equilibrium, if the explicit asymptotic method is supplemented by partial equilibrium approximations designed specifically to deal with the third type of stiffness instability.

Taken together, this paper and the following ones on quasi-steady-state methods [1] and partial equilibrium methods [2] present compelling evidence that algebraically-stabilized explicit integration methods are capable of timesteps competitive with implicit integration methods for a variety of highly-stiff reaction networks. Since explicit methods can execute a timestep faster than an implicit method in a large network, our results suggest that algebraically-stabilized explicit algorithms may be capable of performing as well as, or even substantially outperforming, implicit integration in a variety of moderate to extremely stiff applications. Because of the highly-favorable linear scaling for explicit methods, this fundamentally new view of the efficacy of explicit integration for stiff equations may be particularly important for applications in any field where it is imperative that more realistic—and therefore much larger—networks be used in complex physical simulations.

Acknowledgments

We thank Tony Mezzacappa for useful discussions, Austin Harris for help with some of the calculations, Eric Lingerfelt for programming assistance, and Christian Cardall for a careful reading of the manuscript. Research was sponsored by the Office of Nuclear Physics, U.S. Department of Energy.

[1] Guidry M W and Harris J A 2011 Explicit Integration of Extremely-Stiff Reaction Networks: Quasi-Steady-State Methods (arXiv:1112.4750)
[2] Guidry M W, Billings J J and Hix W R 2011 Explicit Integration of Extremely-Stiff Reaction Networks: Partial Equilibrium Methods (arXiv:1112.4738)
[3] Gear C W 1971 *Numerical Initial Value Problems in Ordinary Differential Equations* (Englewood Cliffs, NJ: Prentice Hall)
[4] Lambert J D 1991 *Numerical Methods for Ordinary Differential Equations* (New York: Wiley)
[5] Press W H, Teukolsky S A, Vettering W T and Flannery B P 1992 *Numerical Recipes in Fortran* (Cambridge: Cambridge University Press)
[6] Oran E S and Boris J P 2005 *Numerical Simulation of Reactive Flow* (Cambridge: Cambridge University Press)
[7] Timmes F X 1999 Integration of Nuclear Reaction Networks for Stellar Hydrodynamics *Astrophys. J. Suppl.* **124**, 241-63
[8] Hix W R and Meyer B S 2006 Thermonuclear kinetics in astrophysics *Nuc. Phys. A* **777**, 188-207
[9] Fryxell B, Olson K, Ricker P, Timmes F X, Zingale M, Lamb D Q, MacNeice P, Rosner R, Truran J and Tufo H 2000 FLASH: An Adaptive Mesh Hydrodynamics Code for Modeling Astrophysical Thermonuclear Flashes *Astrophys. J. Suppl.* **131** 273-334
[10] Rauscher T and Thielemann F-K 2000 Astrophysical Reaction Rates From Statistical Model Calculations *At. Data Nuclear Data Tables* **75** 1-351
[11] Hix W R and Thielemann F-K 1999 Computational methods for nucleosynthesis and nuclear energy generation *J. Comp. Appl. Mathematics* **109** 321-51
[12] Parete-Koon S, Hix W R, Smith M S, Starrfield S, Bardayan D W, Guidry M W and Mezzacappa A 2003 Impact of a new $^{17}\mathrm{F}(p,\gamma)$ reaction rate on nova nucleosynthesis *Astrophysical Journal* **598** 1239-45
[13] Mott D R 1999 New Quasi-Steady-State and Partial-Equilibrium Methods for Integrating Chemically Reacting Systems *PhD thesis* University of Michigan at Ann Arbor
[14] Young T R and Boris J P 1977 A numerical technique for solving ordinary differential equations associated with the chemical kinetics of reactive flow problems *J. Phys. Chem.* **81** 2424-7.
[15] Mott D R, Oran E S and van Leer B 2000 Differential Equations of Reaction Kinetics *J. Comp. Phys.* **164** 407-28
[16] Dahlquist G G 1963 A special stability problem for linear multistep methods *BIT* **3** 27-43
[17] Feger E, Guidry M W and Hix W R 2012 Evaluating Integration Methods for Astrophysical Nuclear Reaction Networks (in preparation)
[18] Feger E 2011 Evaluating Explicit Methods for Solving Astrophysical Nuclear Reaction Networks *PhD thesis* University of Tennessee at Knoxville
[19] Rosswog S, Ramirez-Ruiz E and Hix W R 2008 Atypical thermonuclear supernovae from tidally crushed white dwarfs *Astrophysical Journal* **679** 1385-9